WHAT IS CLAIMED IS:

1. A method comprising:

processing a definition of a function associated with a first language to create description information about the function, the description information being sufficient to enable translation of a call to the function into a call to a corresponding function in a second language without requiring processing of the definition of the function.

2. The method of claim 1, further comprising:

storing the description information in a file of description items.

3. The method of claim 1, wherein processing the definition of the function comprises:

examining the definition of the function associated with the first language;

deriving information about the function; and

using the derived information to translate the call to the function into a call to a corresponding function in the second language.

4. The method of claim 3, further comprising:

using the derived information to create the description information.

5. The method of claim 3, further comprising:

storing the translated function in the second language in a library of entries.

6. The method of claim 1, in which processing the definition of the function comprises:

deriving a number of declared formal inputs to the function.

7. The method of claim 1, in which processing the definition of the function comprises:

deriving a number of declared formal outputs to the function.

8. The method of claim 1, in which processing the definition of the function comprises:

deriving a scope of the function.

9. The method of claim 1, in which processing the definition of the function comprises:

determining whether the function accepts a variable number of arguments.

10. The method of claim 1, in which processing the definition of the function comprises:

determining whether the function returns a variable number of results.

11. A method comprising:

providing a file of description items, each item including description information about a function associated with a first language, the description information being sufficient to enable translation of a call to the function into a call to a corresponding function in a second language without requiring processing of the definition of the function; and

using the file of description items to translate a first program file into a second program file.

12. The method of claim 11, wherein the description information about the function comprises:

a descriptor identifying a declared number of formal inputs to the function.

13. The method of claim 11, wherein the description information about the function comprises:

a descriptor identifying a declared number of formal outputs to the function.

14. The method of claim 11, wherein the description information about the function comprises:

a descriptor identifying a scope of the function.

15. The method of claim 11, wherein the description information about the function comprises:

a descriptor identifying an acceptance of a variable input argument list into the function.

16. The method of claim 11, wherein the description information about the function comprises:

a descriptor identifying a return of a variable output argument list from the function.

17. The method of claim 11, wherein using the file of description items comprises:

for each call to a function in the first program file, retrieving an item from the file of description items;

using the description information in the item to translate the call to the function in the first language into a call to a corresponding function in the second language; and

storing the translated function in the second program file.

18. The method of claim 11, wherein using the file of description items comprises:

generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying an acceptance of a variable input argument list into the function.

19. The method of claim 11, wherein using the file of description items comprises:

generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying a return of a variable output argument list from the function.

20. The method of claim 11, wherein using the file of description items comprises:

generating a call through a normal interface for the function if the description information includes a descriptor identifying a known number of input and output arguments to the function.

21. A method comprising:

providing a library file including functions defined by a first language;

processing the library file to create a function library and a description file, the function library including one or more functions defined by a second language, each function in the function library being a translated version of a function in the library file, and the description file including description information about each function in the library file, the description information being sufficient to enable translation of a call to the function into a call to a corresponding function in the second language without requiring processing of the definition of the function; and

using the description file to translate a program file from the first language into the second language, wherein each call in the program file to a function in the library file is translated into a call to a corresponding function in the second language.

22. The method of claim 21, wherein processing the library file comprises:

translating the call to each function in the library file into a call to a corresponding function in the second language; and

creating a function library including the translated version of each function in the library file.

23. The method of claim 22, further comprising:

examining the definition of each function in the library file;

deriving information about each function; and

using the derived information to translate the call to each function into a call to a corresponding function in the second language.

24. The method of claim 23, further comprising:

using the derived information about each function to create the description information; and

creating a description file including description information about each function in the library file.

25. The method of claim 21, wherein using the description file comprises:

for each call in the program file to a function in the library file, retrieving the description information about the function from the description file; and

using the description information to translate the call to the function in the first language into a call to a corresponding function in the second language.

26. The method of claim 21, wherein using the description file comprises:

generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying an acceptance of a variable input argument list into the function.

27. The method of claim 21, wherein using the description file comprises:

generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying a return of a variable output argument list from the function.

28. The method of claim 21, wherein using the description file comprises:

generating a call through a normal interface for the function if the description information includes a descriptor identifying a known number of input and output arguments to the function.

29. A computer program product, tangibly stored on a computer-readable medium, for creating a data file, the product comprising instructions operable to cause a programmable processor to:

process a definition of a function associated with a first language to create description information about the function, the description information being sufficient to enable translation of a call to the function into a call to a corresponding function in a second language without requiring processing of the definition of the function.

30. The product of claim 29, further comprising instructions operable to cause a programmable processor to:

store the description information in a file of description items.

31. The product of claim 29, wherein processing the definition of the function comprises:

examining the definition of the function associated with the first language;

deriving information about the function; and

using the derived information to translate the call to the function into a call to a corresponding function in the second language.

32. The product of claim 31, further comprising instructions operable to cause a programmable processor to:

use the derived information to create the description information.

33. The product of claim 31, further comprising instructions operable to cause a programmable processor to:

store the translated function in the second language in a library of entries.

34. The product of claim 29, in which processing the definition of the function comprises:

deriving a number of declared formal inputs to the function.

35. The product of claim 29, in which processing the definition of the function comprises:

deriving a number of declared formal outputs to the function.

36. The product of claim 29, in which processing the definition of the function comprises:

deriving a scope of the function.

37. The product of claim 29, in which processing the definition of the function comprises:

determining whether the function accepts a variable number of arguments.

38. The product of claim 29, in which processing the definition of the function comprises:

determining whether the function returns a variable number of results.

39. A product, stored on a machine-readable medium, for translating a program file, the product comprising instructions operable to cause a processor to:

provide a file of description items, each item including description information about a function associated with a first language, the description information being sufficient to enable translation of a call to the function into a call to a corresponding function in a second language without requiring processing of the definition of the function; and

use the file of description items to translate a first program file into a second program file.

40. The product of claim 39, wherein the description information about the function comprises:

a descriptor identifying a declared number of formal inputs to the function.

41. The product of claim 39, wherein the description information about the function comprises:

a descriptor identifying a declared number of formal outputs to the function.

42. The product of claim 39, wherein the description information about the function comprises:

a descriptor identifying a scope of the function.

43. The product of claim 39, wherein the description information about the function comprises:

a descriptor identifying an acceptance of a variable input argument list into the function.

44. The product of claim 39, wherein the description information about the function comprises:

a descriptor identifying a return of a variable output argument list from the function.

45. The product of claim 39, wherein using the file of description items comprises:

for each call to a function in the first program file, retrieving an item from the file of description items;

using the description information in the item to translate the call to the function in the first language into a call to a corresponding function in the second language; and

storing the translated function in the second program file.

46. The product of claim 39, wherein using the file of description items comprises:

generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying an acceptance of a variable input argument list into the function.

47. The product of claim 39, wherein using the file of description items comprises:

generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying a return of a variable output argument list from the function.

48. The product of claim 39, wherein using the file of description items comprises:

generating a call through a normal interface for the function if the description information includes a descriptor identifying a known number of input and output arguments to the function.

49. A computer program product, tangibly stored on a computer-readable medium, for translating function calls, the product comprising instructions operable to cause a programmable processor to:

provide a library file including functions defined by a first language;

process the library file to create a function library and a description file, the function library including one or more functions defined by a second language, each function in the function library being a translated version of a function in the library file, and the description file including description information about each function in the library file, the description information being sufficient to enable translation of a call to the function into a call to a corresponding function in the second language without requiring processing of the definition of the function; and

use the description file to translate a program file from the first language into the second language, wherein each call in the program file to a function in the library file is translated into a call to a corresponding function in the second language.

50. The product of claim 49, wherein processing the library file comprises:

translating the call to each function in the library file into a call to a corresponding function in the second language; and

creating a function library including the translated version of each function in the library file.

51. The product of claim 49, further comprising:

examining the definition of each function in the library file;

deriving information about each function; and

using the derived information to translate the call to each function into a call to a corresponding function in the second language.

52. The product of claim 51, further comprising:

using the derived information about each function to create the description information; and

creating a description file including description information about each function in the library file.

53. The product of claim 49, wherein using the description file comprises:

for each call in the program file to a function in the library file, retrieving the description information about the function from the description file; and using the description information to translate the call to the function in the first language into a call to a corresponding function in the second language.

54. The product of claim 49, wherein using the description file comprises:

generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying an acceptance of a variable input argument list into the function.

55. The product of claim 49, wherein using the description file comprises:

generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying a return of a variable output argument list from the function.

56. The product of claim 49, wherein using the description file comprises:

generating a call through a normal interface for the function if the description information includes a descriptor identifying a known number of input and output arguments to the function.